Отчёт по лабораторной работе №1

дисциплина: Информационная безопасность

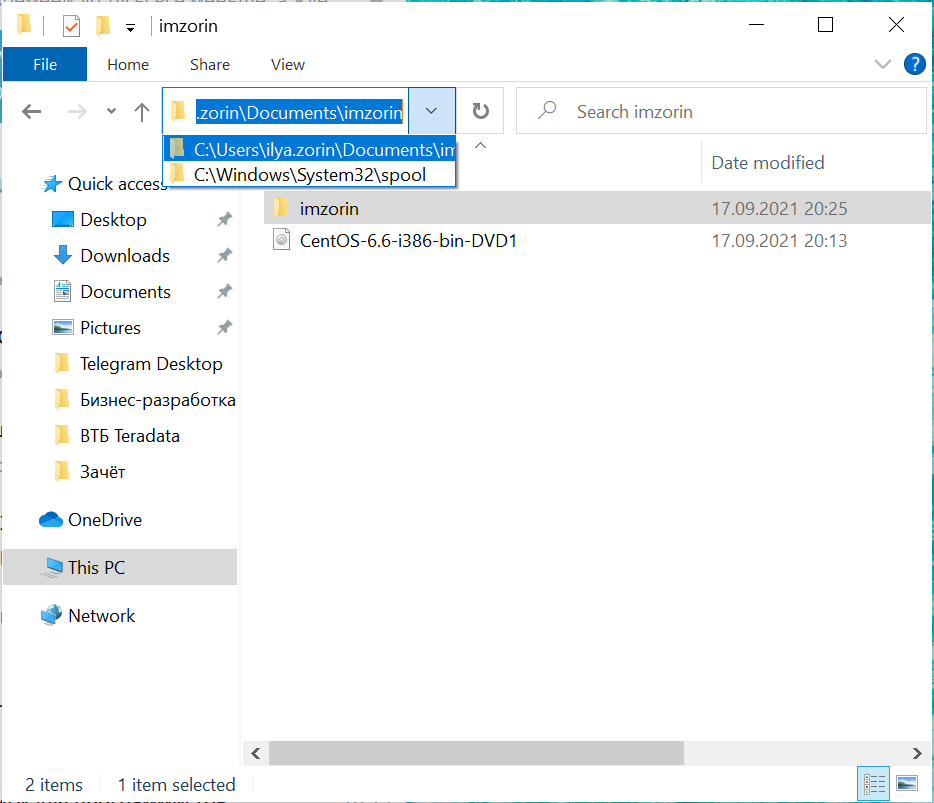
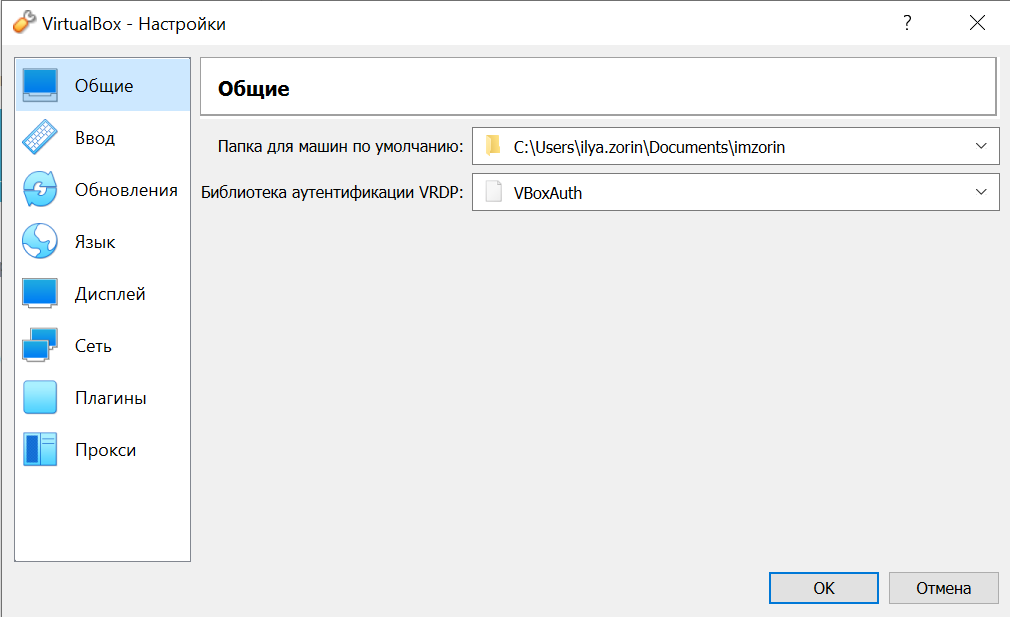
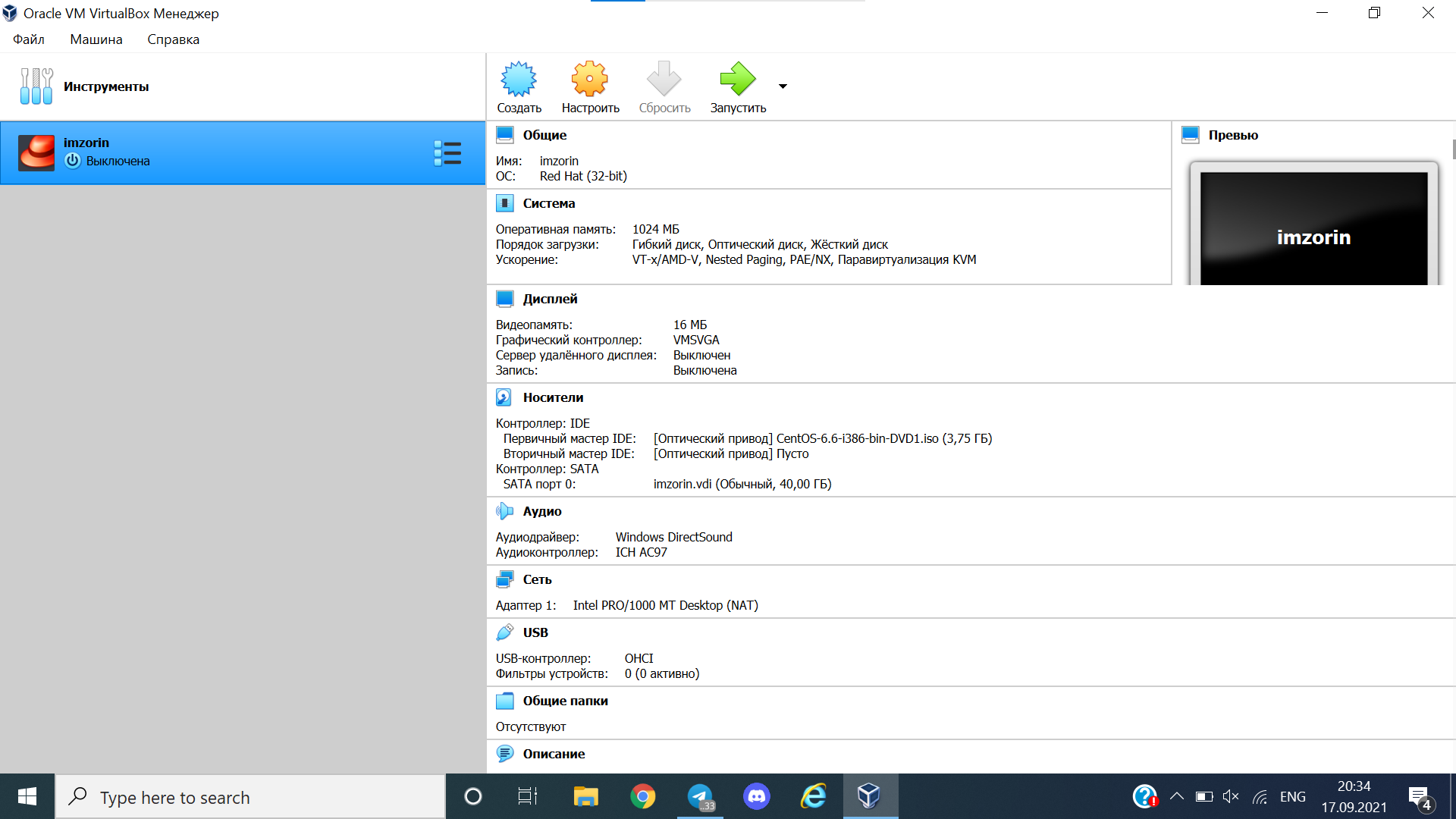
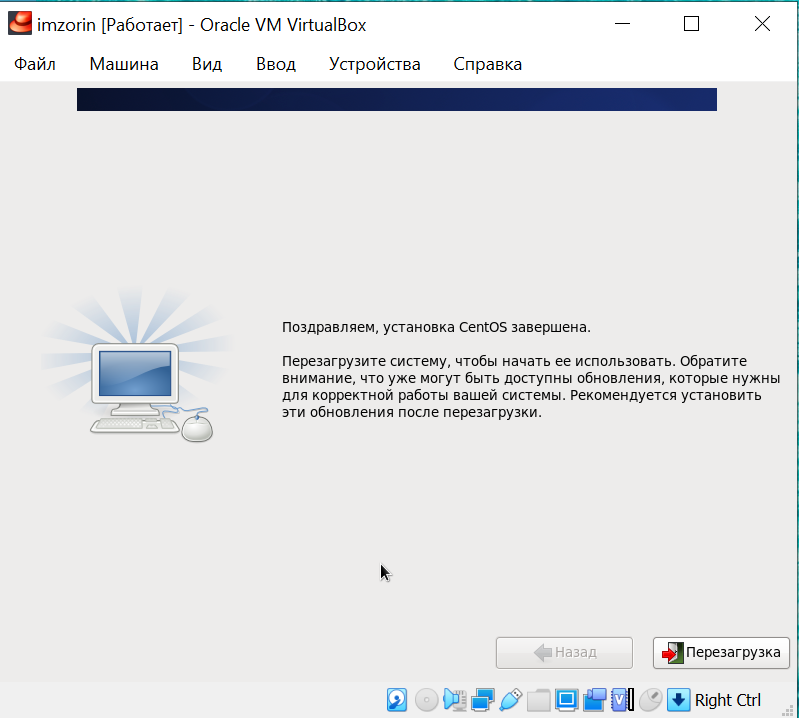
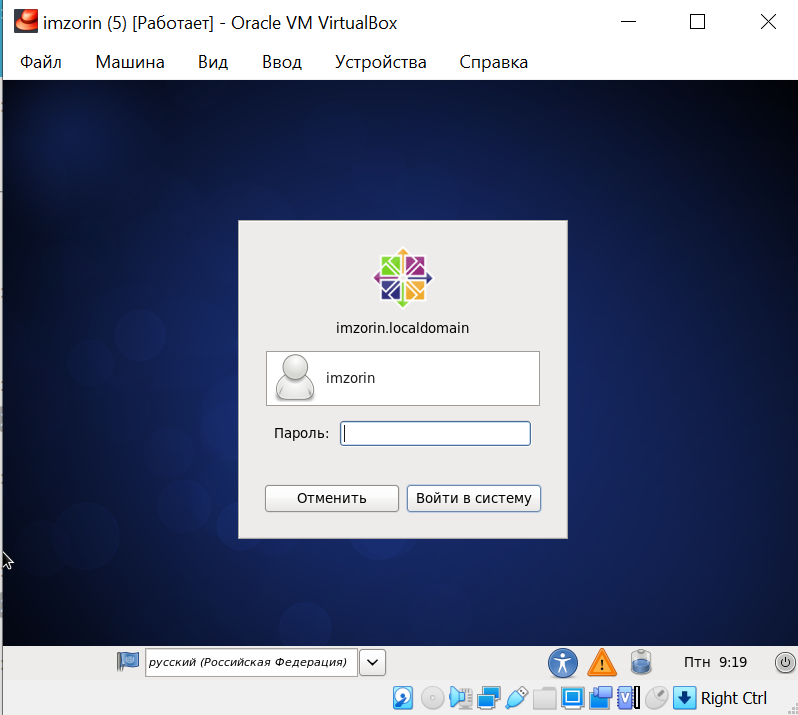
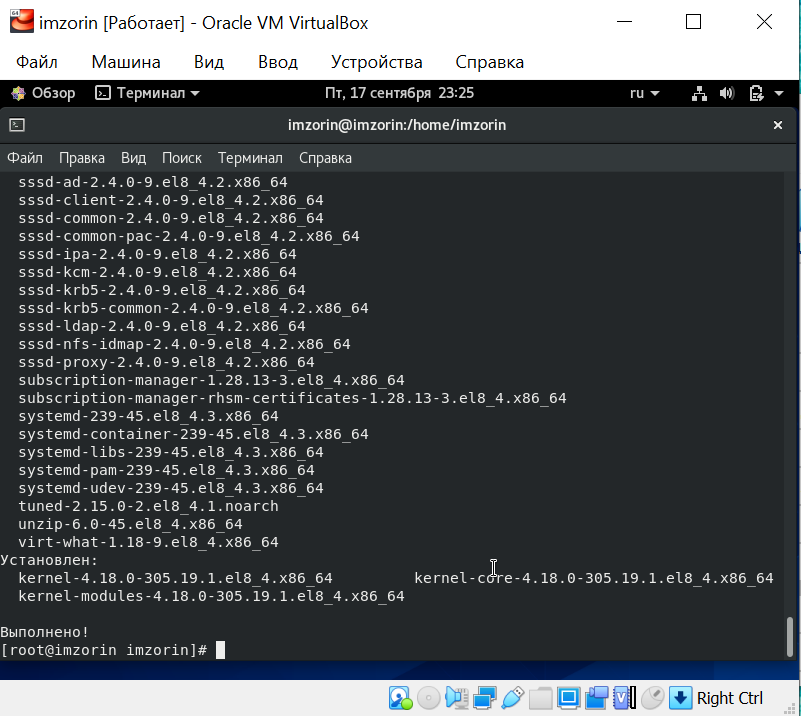
Зорин Илья Михайлович

Содержание

# Цель работы

* Приобретение практических навыков установки операционной системы на виртуальную машину, настройки минимально необходимых для дальнейшей работы сервисов.
* Изучить идеологию и применение средств контроля версий.
* Научиться оформлять отчёты с помощью легковесного языка разметки Markdown.

# Выполнение лабораторной работы

1. Создал именную папку и добавил туда образ вирутальной машины (рис. 1). 
2. Изменил папку по умолчанию для мастоположения каталога виртуальной машины (рис. 2). 
3. Настроил виртуальную машину перед запуском (рис. 3). 
4. Установил ОС (рис. 4). 
5. Подключился с помощью пользователя (рис. 5). 
6. Обновил системные файлы (рис. 6). 

# Выводы

* Приобрёл практические навыки установки операционной системы на виртуальную машину, настройки минимально необходимых для дальнейшей работы сервисов.
* Изучил идеологию и применение средств контроля версий.
* Научился оформлять отчёты с помощью легковесного языка разметки Markdown.

# Ответы на контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются?

* *Это - программное обеспечение для облегчения работы с изменяющейся информацией.*

1. Объясните следующие понятия VCS и их отношения:

* *хранилище - место, где система управления версиями хранит все документы вместе с историей их изменения и другой служебной информацией,*
* *commit - дерево и некая дополнительная информация,*
* *история - данные о версиях файла,*
* *рабочая копия - актулаьная версия файла.*

1. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида.

* *Централизованные VCS имеют модель клиент-сервер: один центральный репозиторий, с которым разработчики взаимодействуют по сети. В отличие от централизованной модели, децентрализованная модель может существовать несколько экземпляров репозитория, которые время от времени синхронизируются между собой.*
* *Пример централизованной VCS - SVN. Одна из самых распространенных систем контроля версий. Для децентрализованной модели примером является git.*

1. Опишите действия с VCS при единоличной работе с хранилищем.

* \*\*

1. Опишите порядок работы с общим хранилищем VCS.

* \*\*

1. Каковы основные задачи, решаемые инструментальным средством git?

* *Системы контроля версий поддерживают возможность отслеживания и разрешения конфликтов, которые могут возникнуть при работе нескольких человек над одним файлом.*
* *Tакже они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви.*

1. Назовите и дайте краткую характеристику командам git.

* *С основными командами можно ознакомиться* [*тут*](https://git-scm.com/book/ru/v2/Приложение-C%3A-Команды-Git-Основные-команды.md)

1. Приведите примеры использования при работе с локальным и удалённым репозиториями.

* *При создании локального репозитория достаточно сначала сделаем предварительную конфигурацию, указав имя и email владельца репозитория:*
* *git config –global user.name “Имя Фамилия”*
* *git config –global user.email “work@mail”*
* *При работе с удалёнными репозиториями необходимо сначала зарегистрироваться на сервере, например, github.*

1. Что такое и зачем могут быть нужны ветви (branches)?

* *Ветвь — направление разработки, независимое от других. Ветвь представляет собой копию части (как правило, одного каталога) хранилища, в которую можно вносить свои изменения, не влияющие на другие ветви. Документы в разных ветвях имеют одинаковую историю до точки ветвления и разные — после неё.*

1. Как и зачем можно игнорировать некоторые файлы при commit?

* *Во время работы над проектом так или иначе могут создаваться файлы, которые не требуется добавлять в последствии в репозиторий. Например, временные файлы, создаваемые редакторами, или объектные файлы, создаваемые компиляторами. Можно прописать шаблоны игнорируемых при добавлении в репозиторий типов файлов в файл .gitignore с помощью сервисов.*